Chapter 2

Parallel Architectures
2.1 Hardware-Architecture

Flynn’s Classification

The most well known and simple classification scheme differentiates according to the multiplicity of instruction and data streams.

The combination yields four classes:

<table>
<thead>
<tr>
<th></th>
<th>SI (single instruction)</th>
<th>MI (multiple instruction)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SD (single data)</td>
<td>SISD: Von-Neumann-Computer</td>
<td>MISD: Data flow machines</td>
</tr>
<tr>
<td>MD (multiple data)</td>
<td>SIMD: Vector computer, (Cray-1, CM2, MasPar, some GPUs)</td>
<td>MIMD: Multiprocessor systems, Distributed Systems (Cray T3E, Cluster, most current computers..)</td>
</tr>
</tbody>
</table>
Flynn’s Classification scheme

SISD

SIMD

MISD

MIMD
a) Multiprocessor systems (*shared memory systems, tightly coupled systems*)

All processors have access to common memory modules via a common communication network.

Further distinction according to the accessibility of the memory modules:

- **uniform memory access (UMA):**
  Access to memory module is independent of address of processor and memory module.

- **non-uniform memory access (NUMA):**
  Memory access depends on addresses involved.
MIMD Machines

b) Multicomputer systems (*message passing systems, loosely coupled systems*)

Each processor has its own private memory with exclusive access. Data exchange takes place by sending messages across an interconnection network.
Pragmatic Classification (Parallel Computer)

Parallel Computer
- Cluster Computer
  - Pile of PCs
    - Beowulf
  - NT/Windows Clusters
- Metacomputer
- Tightly Coupled
- Constellations
- Vector
- WS Farms/cycle-Stealing
- DSM – SHMEM-NUMA
2.2 Interconnection networks

General Criteria

- **Extendibility**
  - Arbitrary increments
- **Performance**
  - Short paths between all processors
  - High bandwidth
  - Short delay
- **Cost**
  - Proportional to number of wires and to number of access points
- **Reliability**
  - Existence of redundant data paths
- **Functionality**
  - Buffering
  - Routing
  - Group communication
Topological Performance Aspects

- **Node degree**
  - Def: Number of directly connected neighbor nodes
  - Goal: constant, small (cost, scalability)

- **Diameter**
  - Def: Maximum path length between two arbitrary nodes
  - Goal: small (low maximum message delay)

- **Edge connectivity**
  - Def: Minimum number of edges to be removed in order to partition the network
  - Goal: high (bandwidth, fault tolerance, parallelism)

- **Bisection bandwidth**
  - Def: Minimum sum of bandwidth of all sets of edges, which partition the network in two equal halves when removed
  - Goal: high (bandwidth, fault tolerance, parallelism)
2.2.1 Static Networks

Connectivity spectrum of static networks

- **Ring**
  - No. of links: $n$
  - Accesses/Processors: 2
  - Diameter: $n/2$

- **Hypercube**
  - No. of links: $n/2 \log_2 n$
  - Accesses/Processors: $\log_2 n$
  - Diameter: $\log_2 n$

- **Completely meshed up**
  - No. of links: $n (n-1)/2$
  - Accesses/Processors: $n-1$
  - Diameter: 1
Grid Structures

4x4-Grid

4x4-Torus

Properties

• Constant node degree
• Extendibility in small increments
• Good support of algorithms with local communication structure (modeling of physical processes, e.g. Laplace heat distribution)
Hypercube

Properties:

- Logarithmic diameter
- Extendibility in powers of 2
- Variable node degree
Trees

Properties:

- Logarithmic diameter
- Extendibility in powers of 2
- Node degree at most 3 (at most 5, respectively)
- Poor bisection bandwidth
- No parallel data paths (binary tree)
Fat Tree

n-ary Tree
Higher Capacity (factor n) closer to the root
removes bottleneck

Connection Machine CM-5
Cube Connected Cycles CCC(d)

d-dimensional Hypercube, where each node is replaced by a ring of size d. Each of these d nodes has two links in the ring and one more to one of the d hypercube dimensions

Properties:
• Logarithmic diameter
• Constant node degree (3)
• Extendibility only in powers of 2
Butterfly-Graph

Properties:
• Logarithmic diameter
• Extendibility in powers of 2
• Constant node degree 4
### Overview: Properties

<table>
<thead>
<tr>
<th>Graph</th>
<th>Number of nodes</th>
<th>Number of edges</th>
<th>Max. Node degree</th>
<th>Diameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Grid ( G(a_1 \times a_2 \times \cdots \times a_d) )</td>
<td>( \prod_{k=1}^{d} a_k )</td>
<td>( \sum_{k=1}^{d} (a_k - 1) \prod_{i \neq k} a_i )</td>
<td>( 2d )</td>
<td>( \sum_{k=1}^{d} (a_k - 1) )</td>
</tr>
<tr>
<td>Torus ( T(a_1 \times a_2 \times \cdots \times a_d) )</td>
<td>( \prod_{k=1}^{d} a_k )</td>
<td>( d \prod_{k=1}^{d} a_k )</td>
<td>( 2d )</td>
<td>( \sum_{k=1}^{d} \lfloor a_k / 2 \rfloor )</td>
</tr>
<tr>
<td>Hypercube ( H(d) )</td>
<td>( 2^d )</td>
<td>( d2^{d-1} )</td>
<td>( d )</td>
<td>( d )</td>
</tr>
<tr>
<td>Cube Connected Cycles ( CCC(d) )</td>
<td>( d2^d )</td>
<td>( 3d2^{d-1} )</td>
<td>( 3 )</td>
<td>( 2d + \lfloor d / 2 \rfloor )</td>
</tr>
</tbody>
</table>
2.2.2 Dynamic Interconnection networks

All components have access to a joint network. Connections are switched on request.

Scheme of a dynamic network

There are basically three different classes

- Bus
- Crossbar switch
- Multistage networks
Bus-like Networks

- cost effective
- blocking
- extendible
- suitable for small number of components

Single bus

Multibus for multiprocessor architecture

Barry Linnert, linnert@inf.fu-berlin.de, Cluster Computing SoSe 2017
Crossbar switch

- expensive
- non-blocking, highly performant
- fixed number of access points
- realizable only for small networks due to quadratically growing costs

Interprocessor connection

Connection between processors and memory modules
2.2.3 Multistage Networks

Small crossbar switches (e.g. 2x2) serve as cells that are connected in stages to build larger networks.

Properties
• partially blocking
• extendible

(a)             (b)             (c)              (d)

Elementary switching states:
through (a), cross (b), upper (c) and lower (d) broadcast
Example: Omega-Network

Target address determines selection of output for each stage (0: up, 1: down)
Example: Banyan-Network
Classes of dynamic interconnection networks with their basic properties in comparison:

<table>
<thead>
<tr>
<th></th>
<th>Bus</th>
<th>Multistage Network</th>
<th>Crossbar switch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latency (distance)</td>
<td>1</td>
<td>(\log n)</td>
<td>1</td>
</tr>
<tr>
<td>Bandwidth per access point</td>
<td>(1/n)</td>
<td>&lt; 1</td>
<td>1</td>
</tr>
<tr>
<td>Switching cost</td>
<td>(n)</td>
<td>(n \log n)</td>
<td>(n^2)</td>
</tr>
<tr>
<td>Wiring cost</td>
<td>1</td>
<td>(n \log n)</td>
<td>(n)</td>
</tr>
</tbody>
</table>

Asymptotic growth of cost and performance features of dynamic interconnection networks
2.2.4 Local Networks

- mostly bus-like or ring-like topologies
- diverse media (e.g. coaxial cable, twisted pair, optical fiber, infrared, radio)
- diverse protocols (e.g. IEEE 802.3 (Ethernet, Fast-Ethernet, Gigabit-Ethernet), IEEE 802.5 (Token-Ring), FDDI, ATM, IEEE 802.11 (Wlan/WiFi), IEEE 802.15 (Bluetooth), IEEE 802.16 (WiMAX))
- compute nodes typically heterogeneous regarding performance, manufacturer, operating system
- mostly hierarchical heterogeneous structure of subnetworks (structured networks)
Typical Intranet

- Local area network
- Web server
- Email server
- File server
- Router/firewall
- Print and other servers
- The rest of the Internet
- Desktop computers
- Print
- Other servers
A Cluster is the collection of usually complete autonomous computers (Workstations, PCs) to build a parallel computer. (Some cluster nodes share infrastructure such as power supply, case, fans).

The component computer nodes can but need not to be spatially tightly coupled.

The coupling is realized by a high speed network. Networks (typically) used:

- Ethernet (Fast, Gigabit, 10 Gigabit, ...)
- Myrinet
- SCI (Scalable Coherent Interface)
- Infiniband
Linux-Cluster (History)

1995  Beowulf-Cluster (NASA)
      PentiumPro/II as nodes, FastEthernet, MPI
1997  Avalon (Los Alamos) 70 Alpha-Processors
1998  Ad-hoc-Cluster of 560 nodes for one night (Paderborn)
1999  Siemens hpcLine 192 Pentium II (Paderborn)
2000  Cplant-Cluster (Sandia National Laboratories)
      1000 Alpha-Prozessoren
2001  Locus Supercluster 1416 Prozessoren
2002  Linux NetworX 2304 Prozessoren (Xeon)
      ....
140 Alpha-Processors
World record 1998:
560 nodes running Linux

WDR-Computer night 1998, HNF / Uni Paderborn

Spontaneously connected heterogeneous cluster of private PCs brought by visitors.

Network: Gigabit-/Fast Ethernet
Supercomputer (Cluster)

- **ASCI White**
- **ASCI Q**
- **ASCI Blue Pacific**

Barry Linnert, linnert@inf.fu-berlin.de, Cluster Computing SoSe 2017
Networks for Cluster

- **Fast Ethernet**
  - Bandwidth: 100 Mb/s
  - Latency: ca. 80 $\mu$sec (userlevel to userlevel)
  - Was highly prevalent (low cost)

- **Gigabit-Ethernet**
  - Bandwidth: 1 Gb/s
  - Latency: ca. 80 $\mu$sec (userlevel to userlevel, raw: 1-12 $\mu$sec)
  - Today widespread (159 of Top500, Nov 2012)

- **10 Gigabit-Ethernet**
  - Bandwidth: 10 Gbit/s
  - Latency: ca. 80 $\mu$sec (userlevel to userlevel, raw: 2-4 $\mu$sec)
  - Already in use (30 of Top500, Nov 2012)
Myrinet

- Bandwidth: 4 Gb/s
- Real: 495 MB/s
- Latency: ca. 5 μsec
- Point-to-Point
- Message Passing
- Switch-based
- Still used (3 in Top 500, Nov 2012, using 10G Myrinet)
Scalable Coherent Interface

- 64-bit global address space (16-bit node ID, 48-bit local memory).
- Guaranteed data transport by packet-based handshake protocol.
- Interfaces with two unidirectional Links that can work simultaneously.
- Parallel copper or serial optical fibers.
- Low latency (<2μs).
- Supports shared memory and message passing.
- Defines Cache coherence protocol (not available everywhere).
Scalable Coherent Interface (SCI)

Standard IEEE 1594
Bandwidth: 667MB/s
Real: 386MB/s
Latency: 1.4 msec
Shared Memory in HW
Usually 2D/3D-Torus
Scarcely used
(last time Nov. 2005 in Top500)
(Data from Dolphin Inc.)
SCI Nodes

1: Request

2: Request Echo

3: Response

4: Response Echo

Barry Linnert, linnert@inf.fu-berlin.de, Cluster Computing SoSe 2017
SCI for PC-Cluster (Dolphin-Adapter)
Example: SCI-Cluster (Uni Paderborn)

- 96 Dual processor-PCs (Pentium III, 850 MHz) with total 48 GB memory, connected as SCI-Rings (16x6-Torus).
- NUMA-Architecture: Hardware access to remote memory
SCI-Cluster (KBS, TU Berlin)

- Linux-Cluster
- 16 Dualprocessor-nodes with
  - 1.7 GHz P4
  - 1 GB memory
  - 2D-Torus SCI (Scalable Coherent Interface)
SCI address translation

Process 1 on node A

- 32 bit virtual address space
- MMU address translation
- 32 bit physical address space
- SCI address translation
- Real memory
- I/O-window

Process 2 on node B

- 48 bit global SCI address space
Infiniband

- New standard for coupling devices and compute nodes (replacement for system busses like PCI)
- Single link: 2.5 GBit/sec
- Up to 12 links in parallel
- Most used today (224 of Top500, Nov 2012, different configurations)

Source: top500.org
Infiniband

- Current technology for building HPC-Cluster
- 2.5 Gbps per Link
- ca. 2 µsec Latency
Virtual Interface Architecture

- To standardize the diversity of high performance network protocols, an industry consortium, consisting of Microsoft, Compaq (now HP) and Intel defined the Virtual Interface Architecture (VIA).
- Central element is the concept of a user-level interface, that allows direct access to the network interface bypassing the operating system.

Examples:
- Active Messages (UCB);
- Fast Messages (UIUC);
- U-Net (Cornell);
- BIP (Univ. Lyon, France);

source: M. Baker
Virtual Interface Architecture

Application

OS Communication Interface
(Sockets, MPI,...)

VI User Agent

user mode
Open/Connect/
Register Memory

kernel mode
Send/Receive

VI Kernel Agent

VI Network Adapter

Barry Linnert, linnert@inf.fu-berlin.de, Cluster Computing SoSe 2017
Virtual Interface Architecture

- **Application**
- **OS Communication Interface** (Sockets, MPI,...)
- **VI User Agent**

**user mode**

**kernel mode**

**Open/Connect/ Register Memory**

**Send/Receive**

**VI Kernel Agent**

**VI Network Adapter**
2.4 Architectural Trends in HPC

- Cost saving by usage of mass products
- Usage of top-of-line standard processor
- Usage of top-of-line standard interconnect
- SMPs/multicore/manycore-processors as nodes
- Assemble such nodes to build arbitrarily large clusters
Top 500: Architecture

With respect to number of systems, TOP500, Nov. 2012, www.top500.org

Barry Linnert, linnert@inf.fu-berlin.de, Cluster Computing SoSe 2017
Notion of "Constellation"

- Nodes of parallel computer consist of many processors (SMP) or multicore processors
- Thus, we have two layers of parallelism: the "Intranode-parallelism" and the "Internode parallelism"
- In the TOP500 terminology there is a distinction between *Cluster* and *Constellation*, depending on what form parallelism is dominant:
  - If a machine has more processor cores per node than nodes at all, it is called *Constellation*.
  - If a machine has more nodes than processor cores per node, than it is called *Cluster*. 
Top 500: Processor Types

Processor Architecture Share Over Time
1993-2009

Scalar
Vector
Others

With respect to number of systems, TOP500, Nov 2012, www.top500.org
Top 500: Coprocessors

With respect to number of systems, TOP500, Nov 2012, www.top500.org

1995  2000  2005  2010

- IBM Cell
- ATI Radeon
- Nvidia Fermi
- N/A

- Clearspeed
- Intel Xeon Phi
- Nvidia Kepler
Top 500: Processor Family

Processor Family Share Over Time
1993-2009

- Power
- Intel EM64T
- Intel IA-32
- MIPS
- Sparc
- PA-RISC
- Cray
- Alpha
- AMD x86_64
- Fujitsu
- NEC
- Intel IA-64
- Intel i860
- Others

Barry Linnert, linnert@inf.fu-berlin.de, Cluster Computing SoSe 2017
Top 500: Operating system families

With respect to number of systems, TOP500, Nov 2012, www.top500.org
HPC-Operating systems
Top 500: Interconnect

With respect to number of systems, TOP500, Nov 2012, www.top500.org

- Gigabit Ethernet
- Crossbar
- Cray Interconnect
- Others
- Infiniband
- N/A
- NUMAlink
- SP Switch
- Quadrics
- Fat Tree
- Proprietary Network
- Myrinet
Moore’s law

Number of transistors per chip doubles (roughly) every 18 months.

## Transistors per processor

<table>
<thead>
<tr>
<th>Processor</th>
<th>Year of introduction</th>
<th>Transistor count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel 4004</td>
<td>1971</td>
<td>2,300</td>
</tr>
<tr>
<td>Intel 8008</td>
<td>1972</td>
<td>3,500</td>
</tr>
<tr>
<td>MOS 6502</td>
<td>1975</td>
<td>3,510</td>
</tr>
<tr>
<td>Zilog Z80</td>
<td>1976</td>
<td>8,500</td>
</tr>
<tr>
<td>Intel 8088</td>
<td>1979</td>
<td>29,000</td>
</tr>
<tr>
<td>Intel 80286</td>
<td>1982</td>
<td>134,000</td>
</tr>
<tr>
<td>Motorola 68020</td>
<td>1984</td>
<td>200,000</td>
</tr>
<tr>
<td>Intel 80386</td>
<td>1985</td>
<td>275,000</td>
</tr>
<tr>
<td>Motorola 68030</td>
<td>1987</td>
<td>273,000</td>
</tr>
<tr>
<td>Intel 80486</td>
<td>1989</td>
<td>1,180,000</td>
</tr>
<tr>
<td>Motorola 68040</td>
<td>1990</td>
<td>~1,200,000</td>
</tr>
<tr>
<td>Processor</td>
<td>Year of introduction</td>
<td>Transistor count</td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>Intel Pentium</td>
<td>1993</td>
<td>3,100,000</td>
</tr>
<tr>
<td>Intel Pentium Pro</td>
<td>1995</td>
<td>5,500,000</td>
</tr>
<tr>
<td>AMD K6</td>
<td>1997</td>
<td>8,800,000</td>
</tr>
<tr>
<td>AMD K7</td>
<td>1999</td>
<td>22,000,000</td>
</tr>
<tr>
<td>Intel Pentium 4</td>
<td>2000</td>
<td>42,000,000</td>
</tr>
<tr>
<td>AMD K8</td>
<td>2003</td>
<td>105,900,000</td>
</tr>
<tr>
<td>AMD K8 Dual Core</td>
<td>2005</td>
<td>243,000,000</td>
</tr>
<tr>
<td>IBM Power 6</td>
<td>2007</td>
<td>789,000,000</td>
</tr>
<tr>
<td>AMD Opteron (6 core)</td>
<td>2009</td>
<td>904,000,000</td>
</tr>
<tr>
<td>Intel Xeon EX (10 core)</td>
<td>2011</td>
<td>2,600,000,000</td>
</tr>
<tr>
<td>Intel Itanium (8 core)</td>
<td>2012</td>
<td>3,100,000,000</td>
</tr>
</tbody>
</table>
Xeon EX (Westmere EX) with 10 Cores

Source: computerbase.de
Intel Single Chip Cloud Computer: 48 Cores

Source: Intel
Performance of Top 500

Performance Development

Performance

100 MFlop/s
100 GFlop/s
10 GFlop/s
1 TFlop/s
10 TFlop/s
100 TFlop/s
1 PFlop/s
10 PFlop/s
100 PFlop/s
1 EFlop/s
10 EFlop/s
100 EFlop/s

Lists

Sum
#1
#500

Top 500: Forecast

Projected Performance Development

- Performance scales from 100 MFlop/s to 10 EFlop/s
- Lists from 1995 to 2020
- Lines represent Sum, #1, and #500

Barry Linnert, linnert@inf.fu-berlin.de, Cluster Computing SoSe 2017
Applications should be supported in a transparent way, i.e., we need a software infrastructure that hides the distribution to a high degree.
Two Architectural Approaches for Distributed Systems

**Distributed Operating System**
The operating system itself provides the necessary functionality

**Network Operating System**
The local operating systems are complemented by an additional layer (Network OS, Middleware) that provides distributed functions
General OS Architecture (Microkernel)

Control

OS user interface

Application

Operation and management of real and logical resources

Services

Kernel interface

OS-Kernel

Process management interaction

Kernel area

Process area

Hardware
Architecture of local OS

User processes

OS-Processes

Kernel interface

OS kernel

Hardware
Architecture of distributed OS

User processes

Node boundary

OS-Processes

Interconnect

OS-Kernel

Hardware

Hardware

Barry Linnert, linnert@inf.fu-berlin.de, Cluster Computing SoSe 2017
Process communication

Communication object

Send(C,..)  Receive(C,..)

Kernel

Barry Linnert, linnert@inf.fu-berlin.de, Cluster Computing  SoSe 2017
To overcome the node boundaries in distributed systems for interaction, we need communication operations that are either integrated in the kernel (kernel federation) or as a component outside the kernel (process federation).

When using the kernel federation the "distributedness" is hidden behind the kernel interface. Kernel calls can refer to any object in the system regardless of its physical location. The federation of all kernels is called the federated kernel.

The process federation leaves the kernel interface untouched. The local kernel is not aware of being part of a distributed system. This way, existing OS can be extended to become distributed OS.
Kernel federation

Node boundaries

Process

Comm. software
loc. OS-Kernel

Federated kernel

Comm. software
loc. OS-Kernel

Process
Process federation

Node boundaries

Process

Comm. software

OS-Kernel

Comm. software

OS-Kernel
<table>
<thead>
<tr>
<th></th>
<th>Multiprocessor-OS</th>
<th>Distributed OS</th>
<th>Network-OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>All nodes have the same OS?</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>How many copies of OS?</td>
<td>1</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>Shared ready queue?</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Communication</td>
<td>Shared memory</td>
<td>Message exchange</td>
<td>Message exchange / Shared files</td>
</tr>
</tbody>
</table>

Barry Linnert, linnert@inf.fu-berlin.de, Cluster Computing  SoSe 2017
Transparency I

- Transparency is the property that the user (almost) does not realize the distributedness of the system.
- Transparency is the main challenge when building distributed systems.
  - Access transparency
  - Access to remote objects in the same way as to local ones
  - Location transparency
    - Name transparency
      Objects are addressed using names that are independent of their locations
    - User mobility
      If the user changes his location he can address the objects with the same name as before.
  - Replication transparency
    If for performance improvement (temporary) copies of data are generated, the consistency is taken care of automatically.
Transparency II

- Failure transparency
  Failures of components should not be noticeable.
- Migration transparency
  Migration of objects should not be noticeable.
- Concurrency transparency
  The fact that many users access objects from different locations concurrently must not lead to problems or faults.
- Scalability transparency
  Adding more nodes to the system during operation should be possible. Algorithms should cope with the growth of the system.
- Performance transparency
  No performance penalties due to uneven load distribution.
References


